

Comparing GPU Implementations of Bilateral and Anisotropic Diffusion Filters for 3D Biomedical Datasets

Mark Howison

Visualization Group
Lawrence Berkeley National Lab

SIAM Conference on Imaging Science '10, April 14, 2010



Objectives

Harness the computational power of emerging architectures (e.g. GPUs)

Find the best filter and optimal parameters for denoising a real (biomedical) dataset

Why use GPUs?

Old Trend:

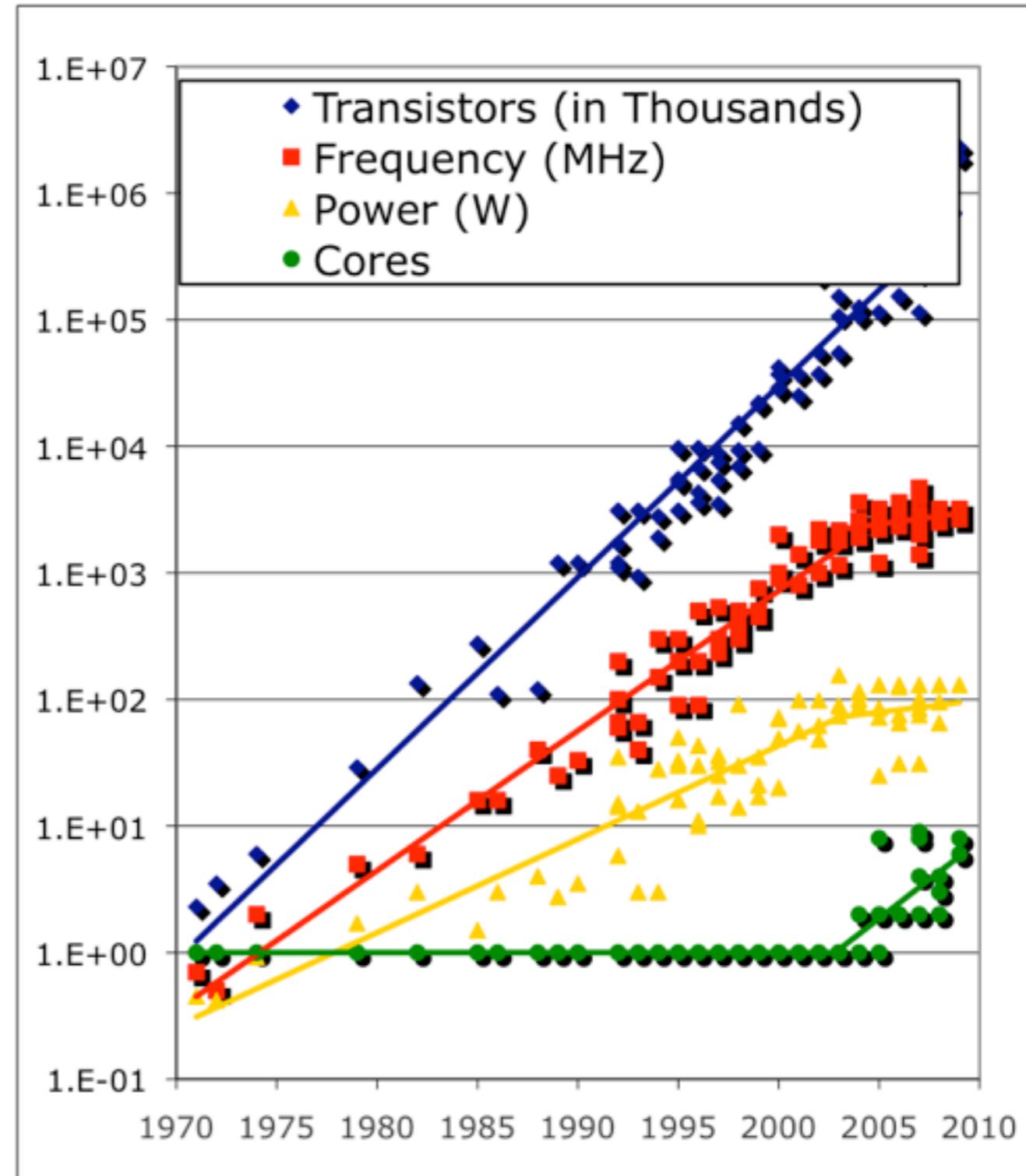
Clock speeds double every 18 months

New Trend:

Number of cores will double instead

Clock speed may decrease

GPUs use many cores to achieve large FLOPs... for the right problems



Courtesy of Kathy Yelick

Parallelism

Task parallelism (multicore)

Serial program: `gcc`

Multiple independent tasks: `gcc *.c`

Execute concurrently: `make -j4`

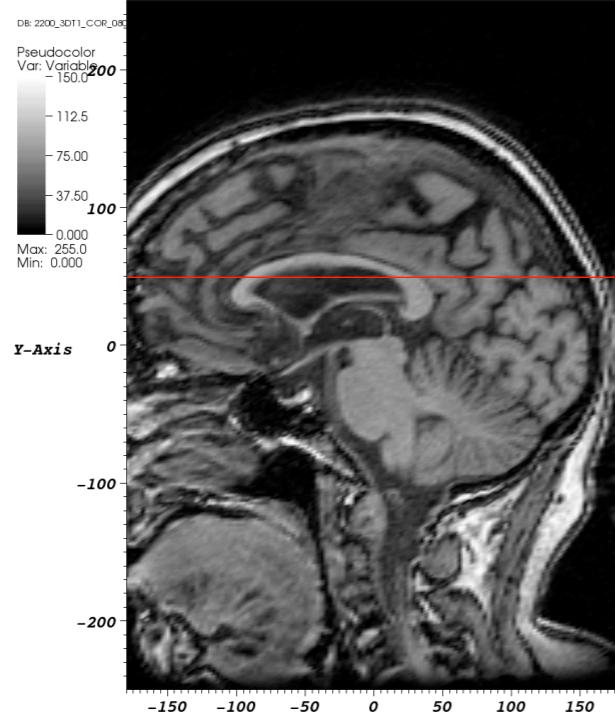
Data parallelism (manycore, GPU)

Array of elements: `image`

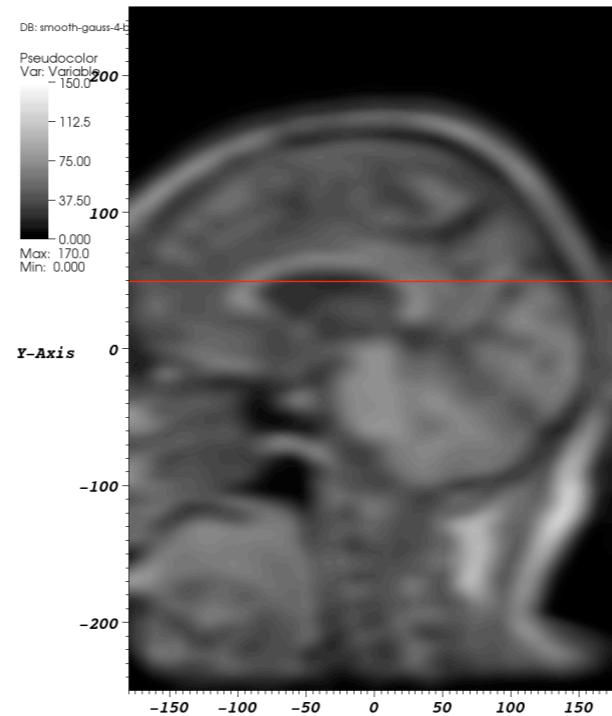
Define an operation on elements: `stencil/kernel`

Apply operation to entire array: `filter`

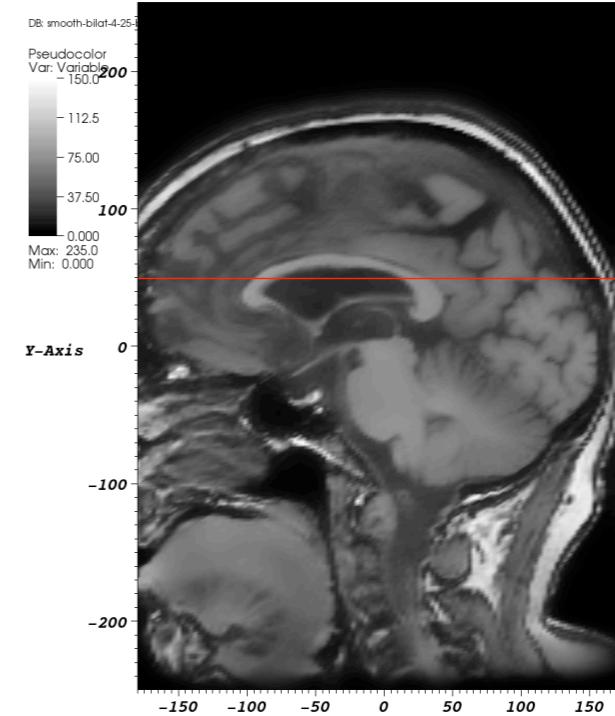
What do these filters do?



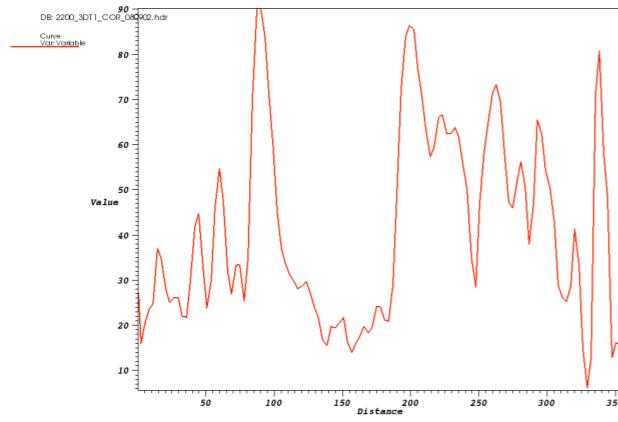
(a) Slice of raw data.



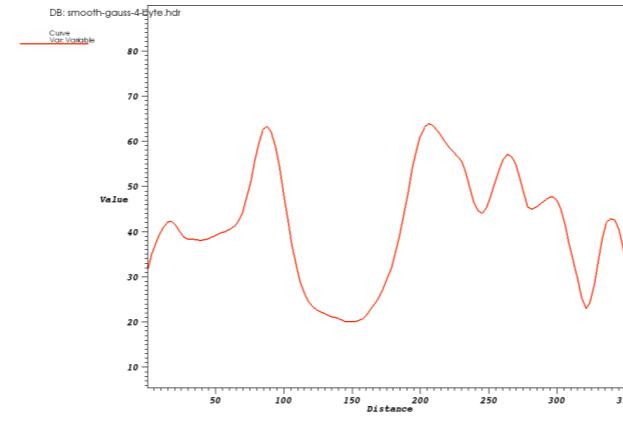
(b) Slice of 3D Gaussian filtered data.



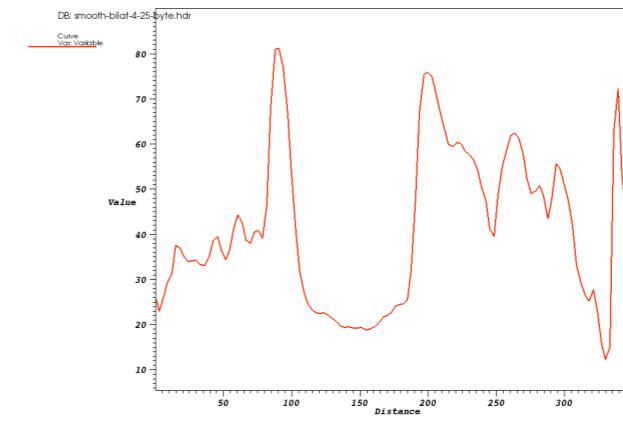
(c) Slice of 3D bilateral filtered data.



(d) Plot of one row of raw data.



(e) Plot of one row of 3D Gaussian filtered data.



(f) Plot of one row of 3D bilateral filtered data.

Edge-preserving smoothing

Implementation

Bilateral filter

Tomasi & Manduchi (1998)

**output is weighted by Gaussians in both
the photometric range + spatial domain**

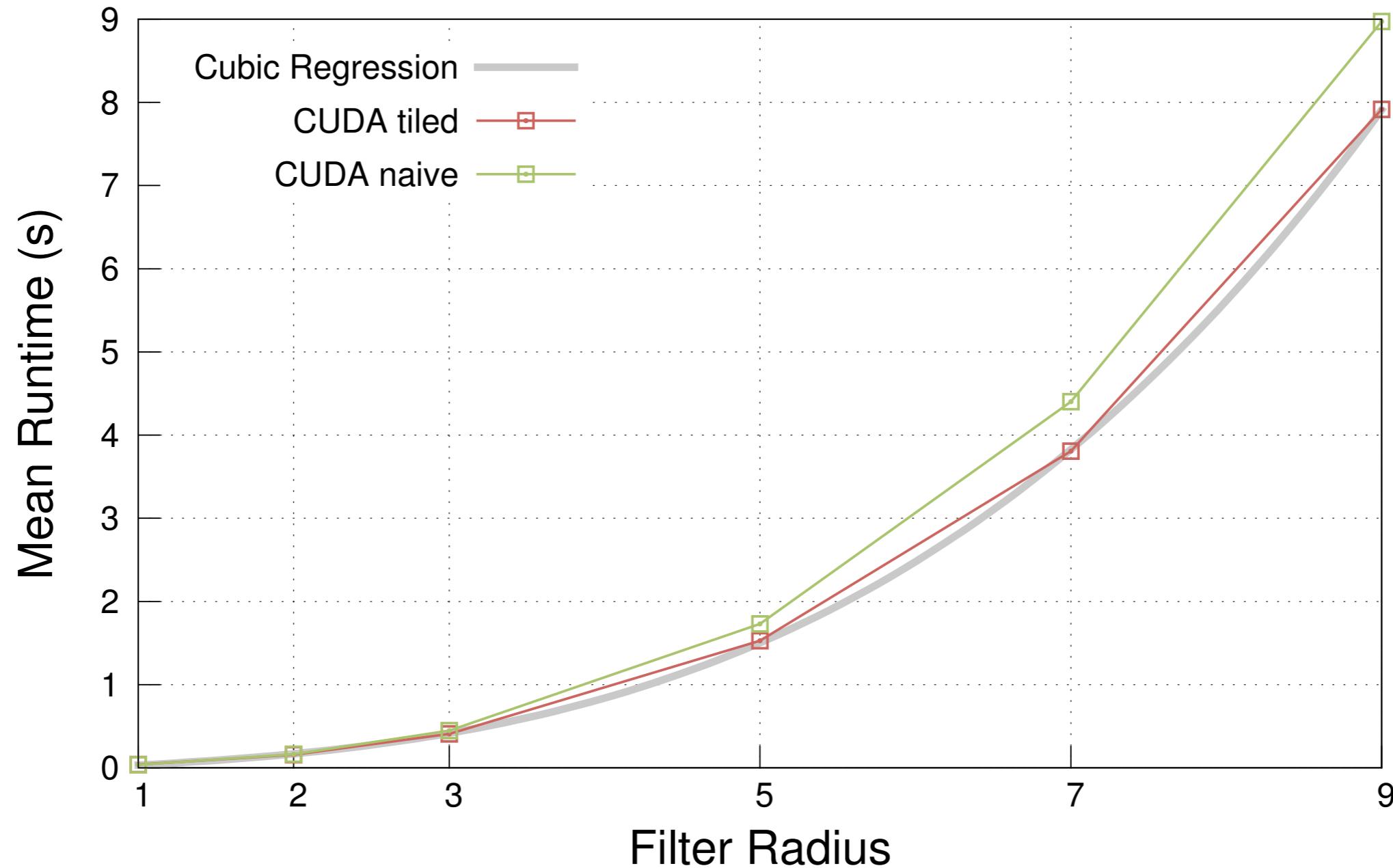
3 parameters

r: filter half-radius

σ_r : standard deviation in photometric range

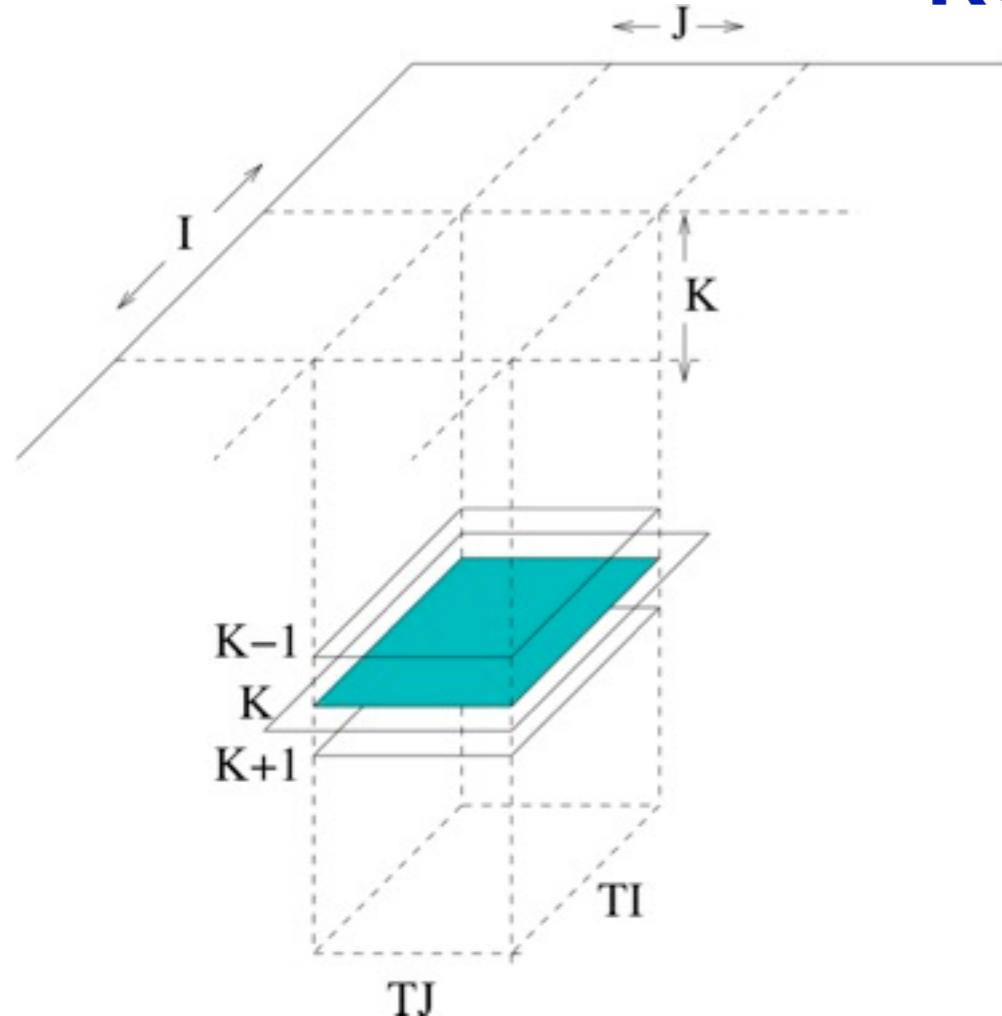
σ_d : standard deviation in spatial domain

Bilateral runtime



Memory tiling in CUDA

Re-uses 3D stencil data



Rivera & Tseng (2000)
use shared memory (16KB per
thread block) in CUDA
experimented (by hand) with
block sizes: found that
8 x 8 x 1 worked well
... could perform an automated
parameter sweep

Implementation

Anisotropic diffusion

solves a diffusion PDE with a “conductance” function that regulates diffusion

chose 3 discrete schemes

Perona & Malik (1990): grid scheme

Gerig et al. (1992): add diagonals to grid

Weickert (1997): precompute gradients

Implementation

Anisotropic diffusion

chose 4 conductance functions

two from Perona & Malik

the “Charbonnier” from Voci et al. (2004)

the “Tukey biweight norm” from Black et al. (1999)

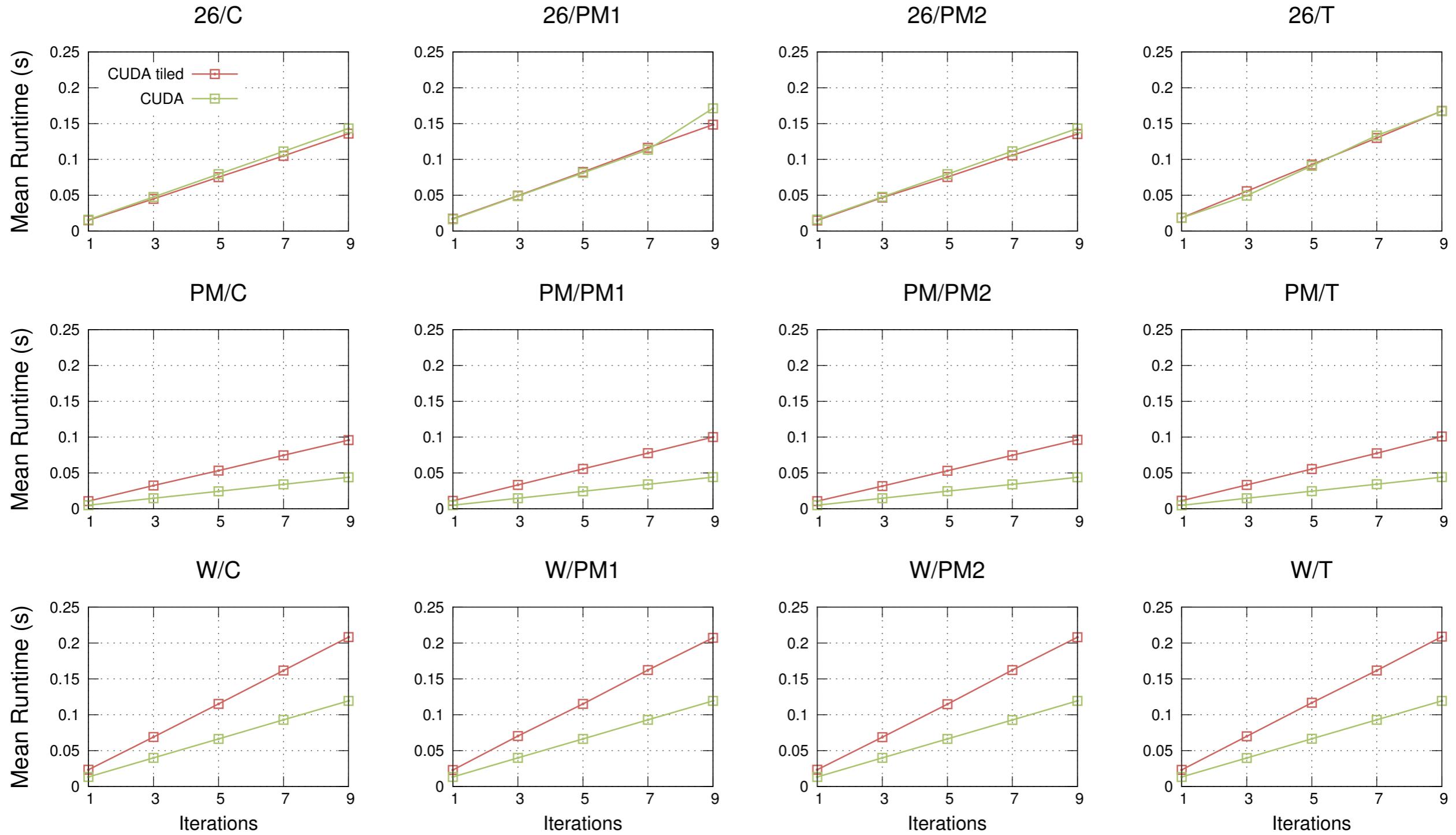
3 other parameters

Δt : discrete timestep

N: number of iterations

k: scale of the conductance function

Anisotropic diffusion runtime



Comparison with serial

Serial

1 x 2.7GHz core (AMD 8384)

2.7GLOPs (without SIMD)

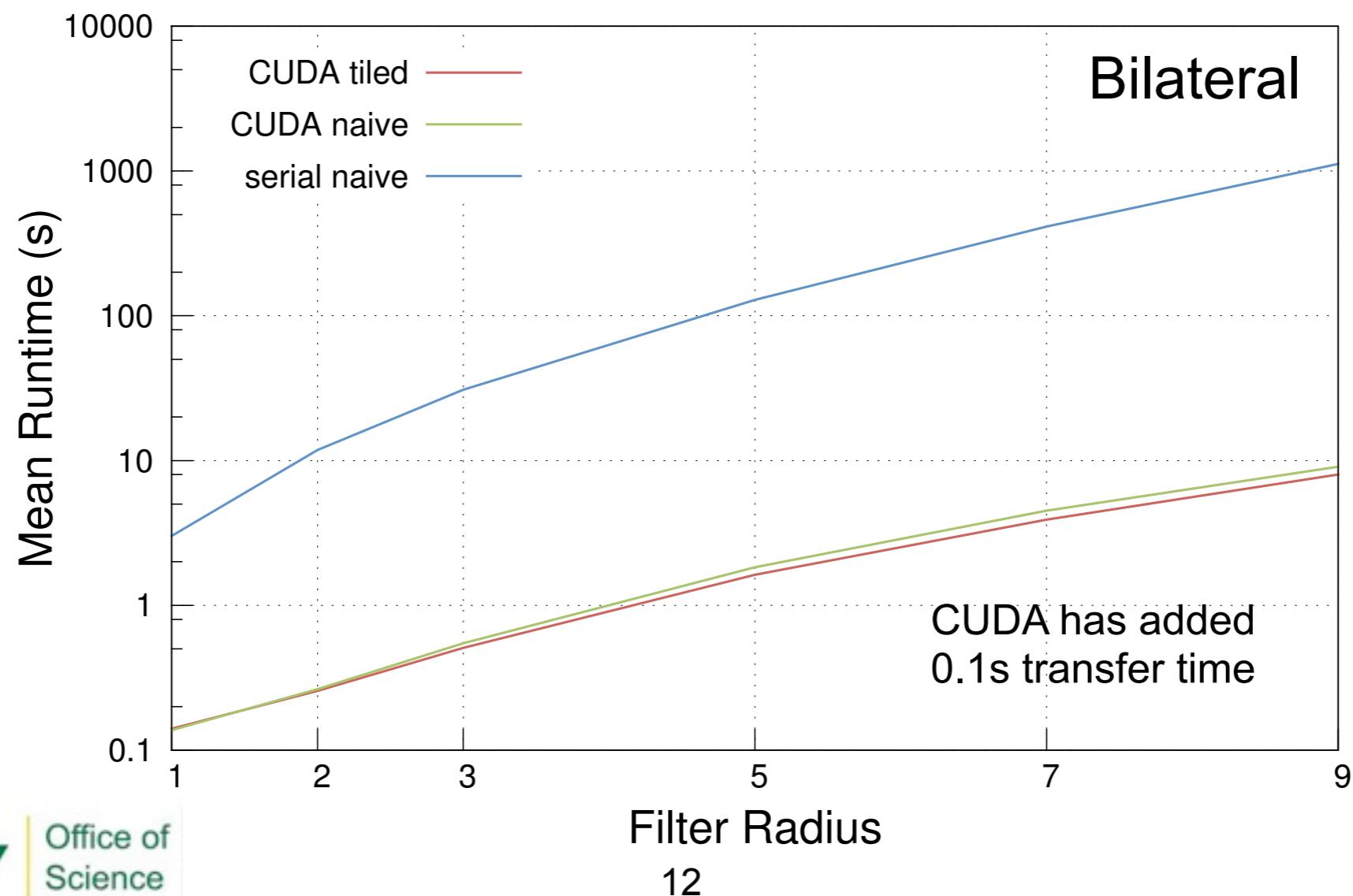
21GB/s memory bandwidth

CUDA

240 x 1.3GHz ‘cores’ (GT200)

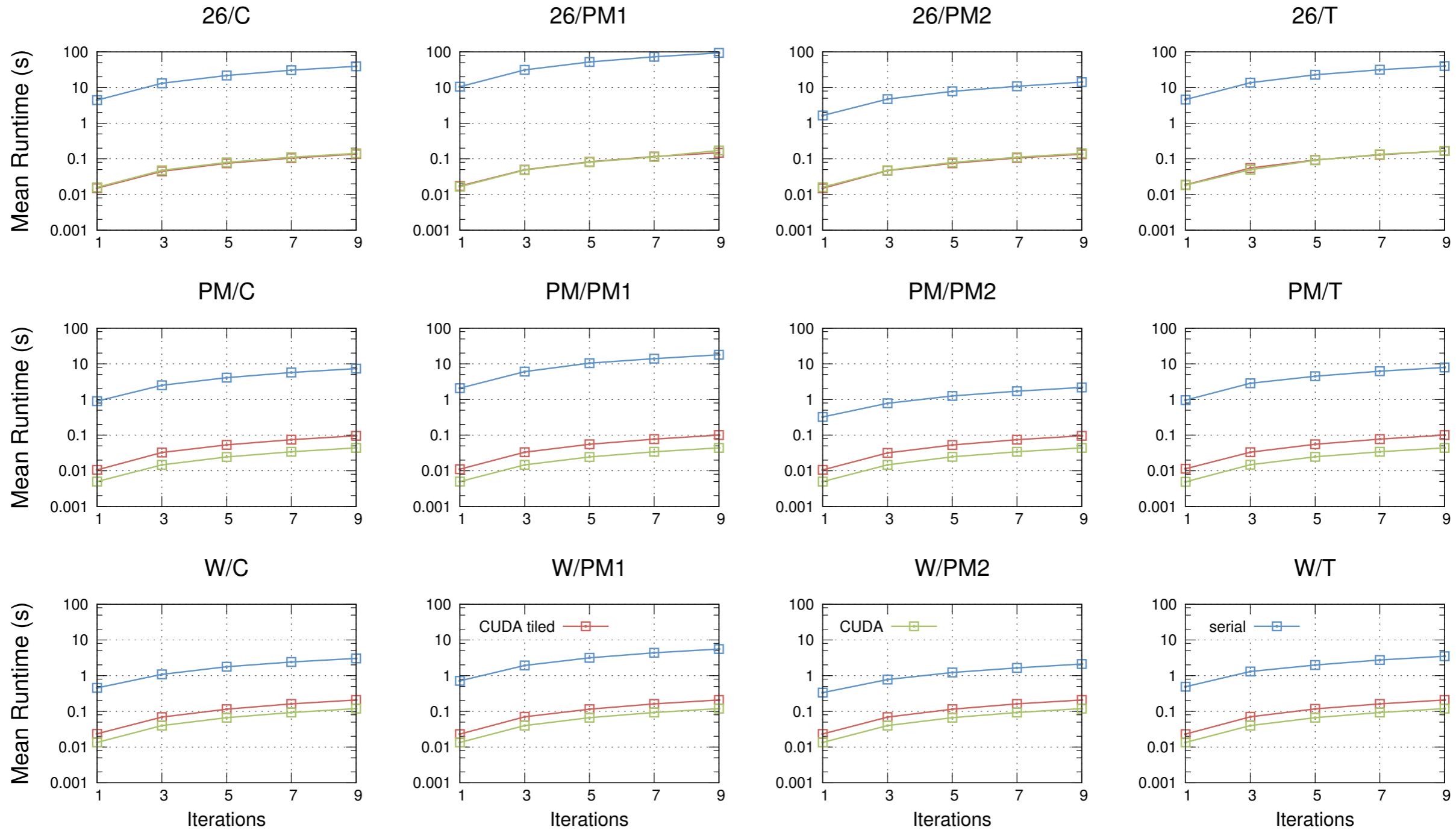
622GFLOPs (without FMAD)

102GB/s memory bandwidth



Comparison with serial

Anisotropic diffusion



Measuring noise

BrainWeb

(<http://mouldy.bic.mni.mcgill.ca/brainweb/>)

reference MR image + noisy image

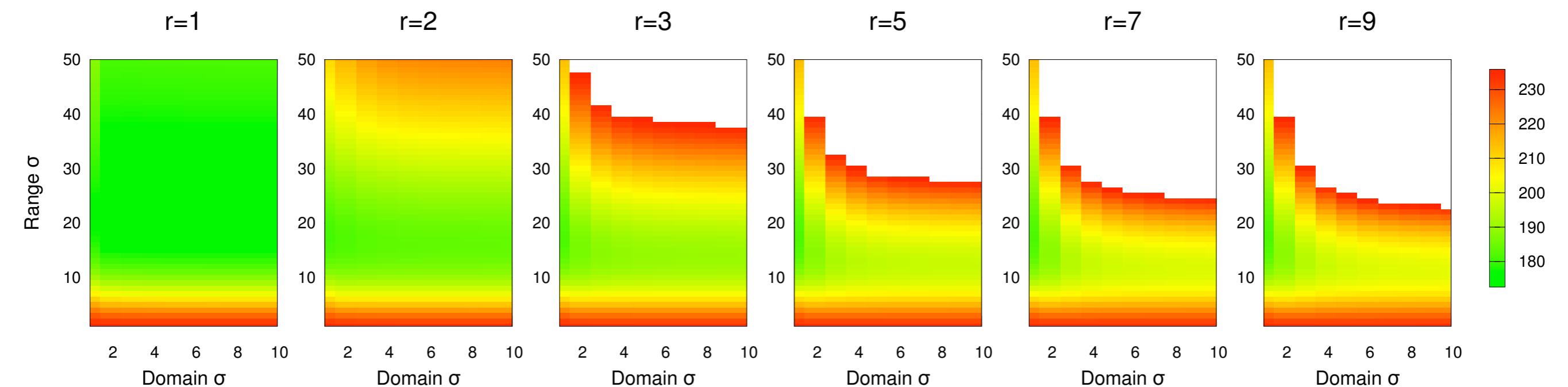
can compute mean squared error

**add up the squares of the distance between
reference and noisy image for all voxels**

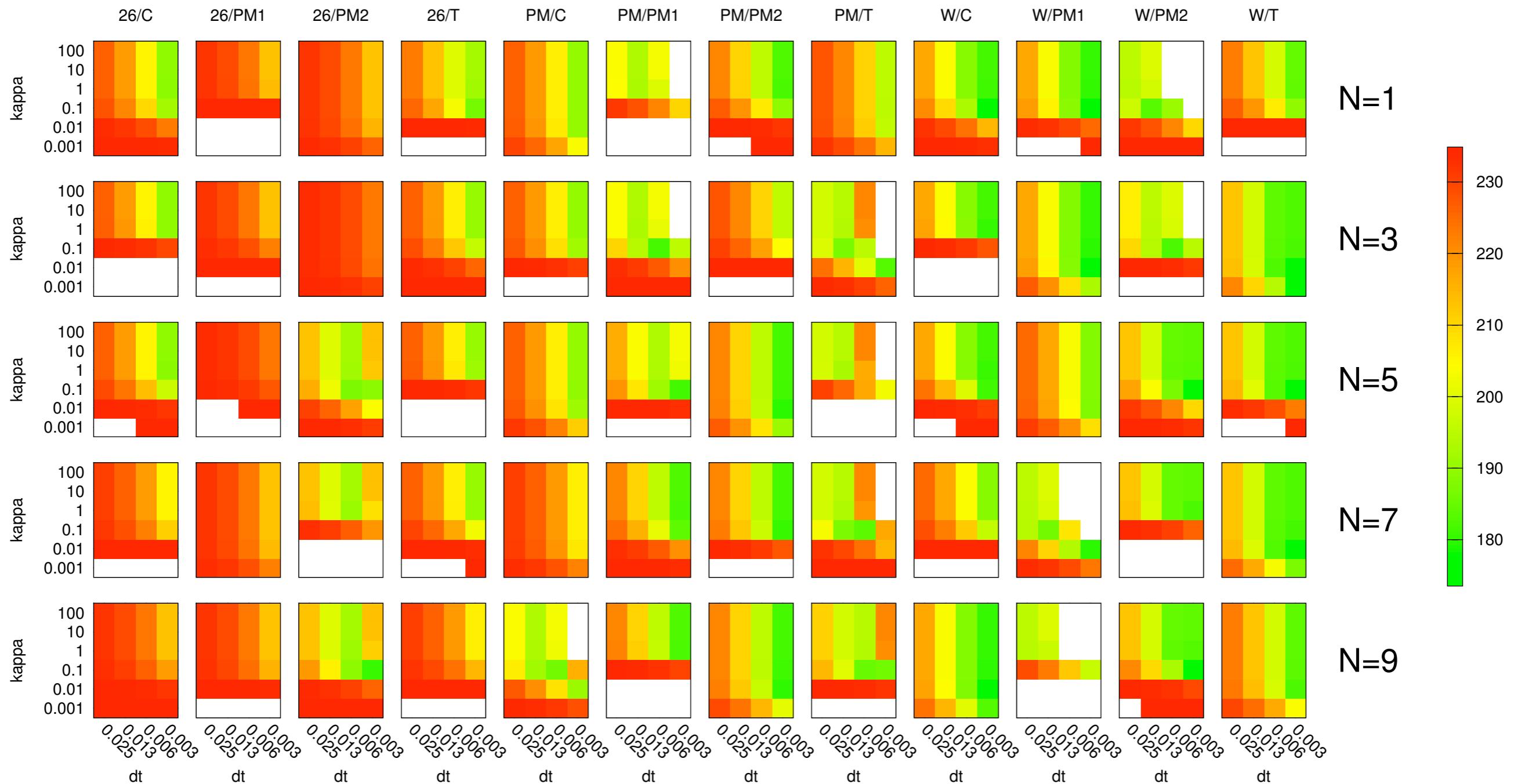
181 x 217 x 181 (=7,109,137 voxels)



Bilateral MSE



Anisotropic diffusion MSE



Lowest MSE

TABLE I
LOWEST MSE FOR BILATERAL FILTER

r	Domain σ	Range σ	MSE	Runtime (s)
1	4	23	172.55	0.040
1	3	23	172.56	0.041
1	5	23	172.56	0.041

TABLE II
LOWEST MSE FOR ANISOTROPIC DIFFUSION

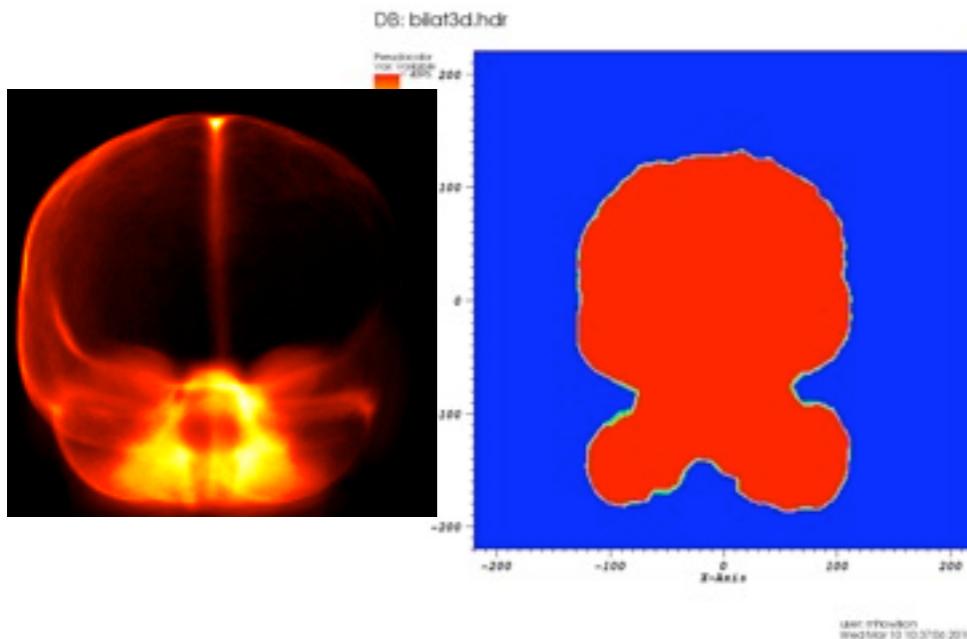
N	Δt	κ	Cond.	Scheme	MSE	Runtime (s)
9	0.025	0.01	PM2	Weickert	173.52	0.206
9	0.025	0.1	PM1	Weickert	174.15	0.207
7	0.025	0.01	PM2	Weickert	174.74	0.160

TABLE III
BEST MSE REDUCTION VS. RUNTIME FOR ANISOTROPIC DIFFUSION

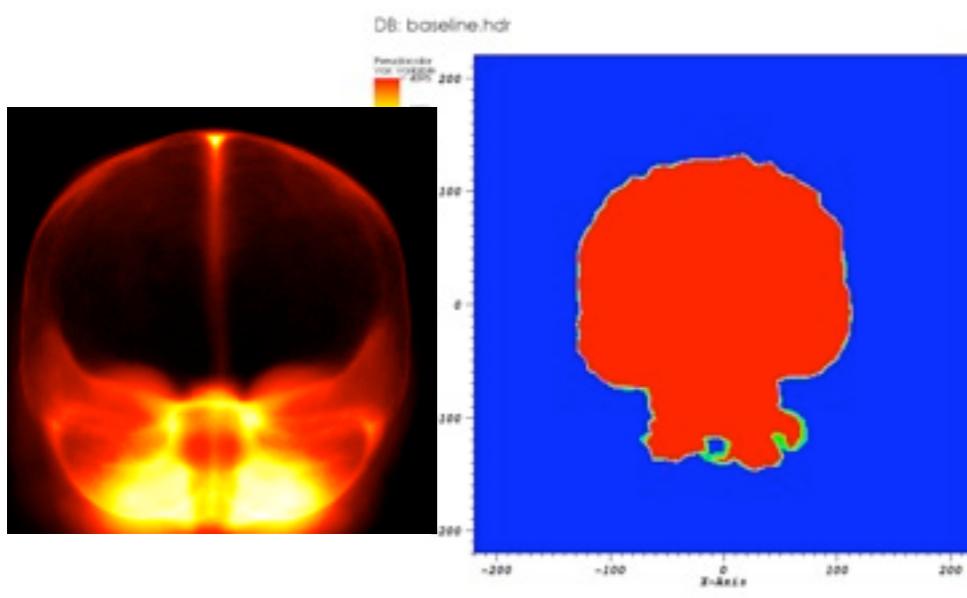
N	Δt	κ	Cond.	Scheme	MSE	Runtime (s)
1	0.025	1	PM2	26-Point	188.66	0.015
1	0.025	1	Char	26-Point	188.87	0.015
1	0.025	10	PM2	26-Point	189.12	0.015

Future work

BET



denoise + BET



Segmentation Validation Engine

(<http://sve.loni.ucla.edu/>)

provides 40 MR images and calculates a score for your segmentation of them

want to measure effectiveness of denoising before segmentation

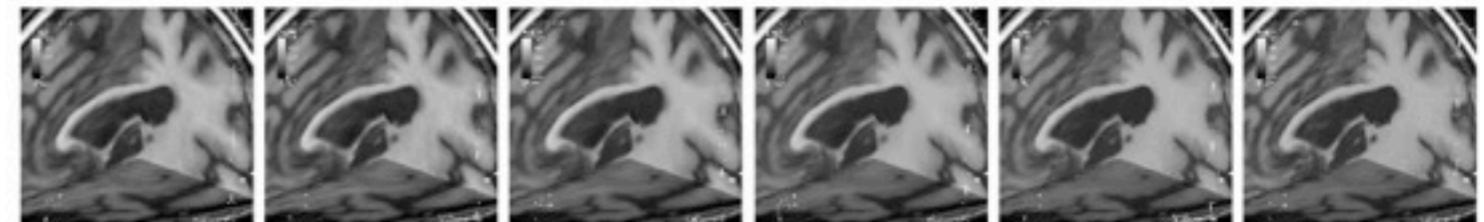
tried using Brain Extraction Toolkit (BET) for segmentation: worse with denoising

... future work

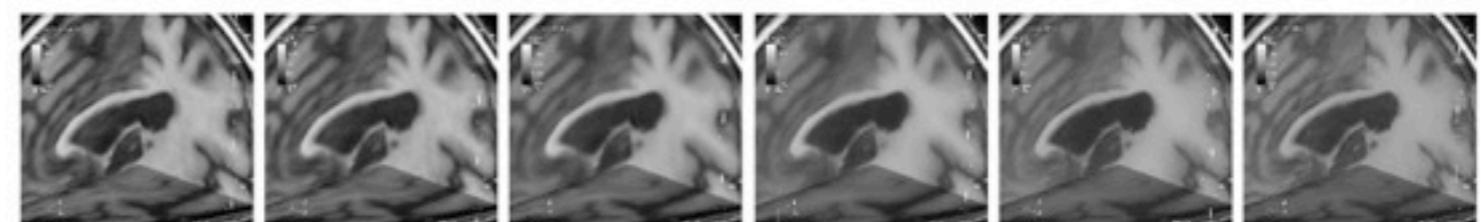
Battery of filtering ->

Interactive exploration

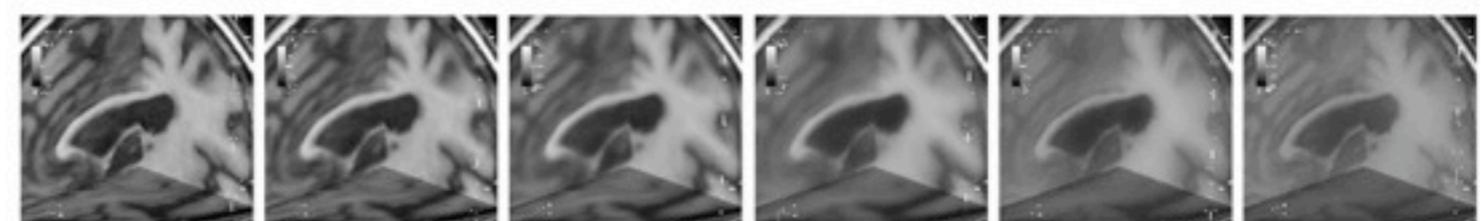
Possible applications



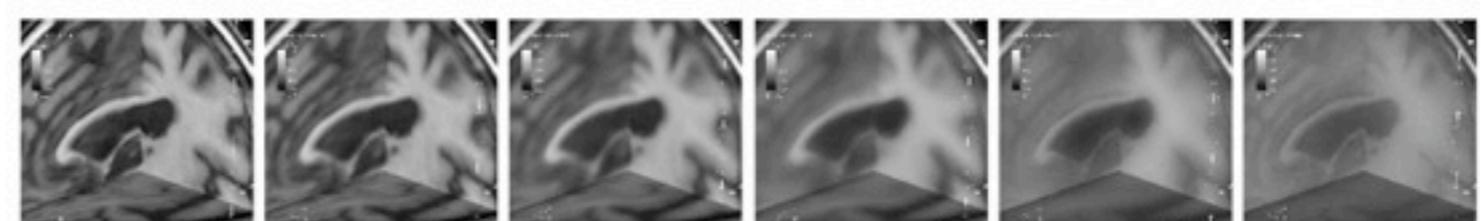
(a) 3D Bilateral smoothing. Left to right: source data, filter radius = 1, 2, 4, 8, 16; photometric difference $\sigma = 5\%$.



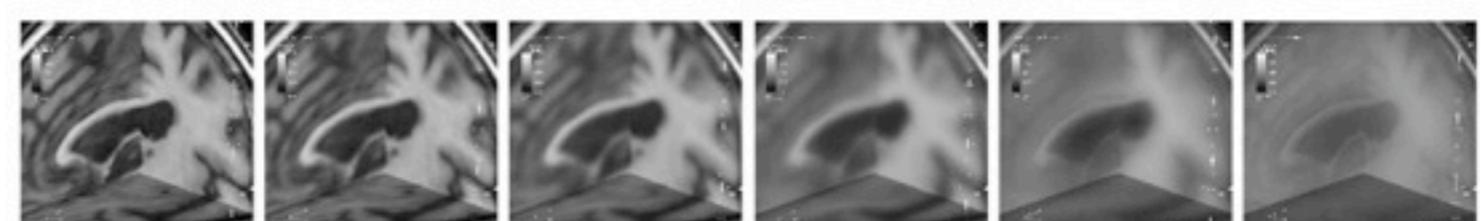
(b) 3D Bilateral smoothing. Left to right: source data, filter radius = 1, 2, 4, 8, 16; photometric difference $\sigma = 10\%$.



(c) 3D Bilateral smoothing. Left to right: source data, filter radius = 1, 2, 4, 8, 16; photometric difference $\sigma = 15\%$.



(d) 3D Bilateral smoothing. Left to right: source data, filter radius = 1, 2, 4, 8, 16; photometric difference $\sigma = 20\%$.



(e) 3D Bilateral smoothing. Left to right: source data, filter radius = 1, 2, 4, 8, 16; photometric difference $\sigma = 25\%$.

Conclusion

CUDA can provide $O(100)$ speedup over serial
Many parameter choices lead to oversmoothing
(according to MSE)
Bilateral filter at $r=1$ had best noise reduction;
anisotropic diffusion at $N=1$ is within 10%

Questions?

mhowison@lbl.gov

Acknowledgment:

Sample data for this work was provided by the UC Davis Alzheimer's Disease Center, which is supported by the National Institutes of Health under grant No. P30 AG010129.